Fire Technology, Vol. 34, No. 3, 1998

A Comparison of Two Methodologies in HAZARD I Fire Egress Analysis

Michael M. Kostreva and *Laura C. Lancaster*
Clemson University, South Carolina

Abstract

Within the framework of HAZARD I, fire egress analysis is performed using the EXITT program. Yet another way to analyze fire egress employs newly developed multiple objective dynamic programming. This paper compares these two approaches by applying them to a model fire of moderate power in a residential building. The findings demonstrate that multiple objective dynamic programming can compute all the paths EXITT finds, but EXITT can't find all the paths multiple objective programming can. Some trade-offs inherent in choosing among the computed egress paths are discussed, and the features of the two fire egress methods are contrasted.

Introduction

The mathematical modeling of fire and related phenomena is of great interest and importance today, with great advances being made by mathematicians, physicists, and other scientists in modeling not only fire behavior but also the smoke and toxins fires produce. These advances have opened many research opportunities, including the study of fire egress, that is, of people trying to escape from a burning building. Some of the questions researchers consider when studying fire egress include how different people react psychologically to fire and how those differences can be determined and defined; where the shortest or the safest path out of the building can be found, taking into account the properties of the fire and its effects on the building; what levels of toxins and heat are lethal or disabling; and how egress paths can be used to minimize such levels.

To model fire hazards in residential buildings, the National Institute of Standards and Technology (NIST), an agency of the U.S. Government, has developed software called HAZARD I,[2] which allows the computer user to define a building, locate its occupants, and choose the context of the fire. HAZARD I also simulates the fire and reports its consequences together with occupants' responses. Of interest in this fire egress problem is finding the set of optimal routes out of the building for each occupant under different fire scenarios.

The EXITT program of HAZARD I determines a unique route out of the building for each occupant by applying several decision rules, one of which is finding the shortest path out of the building. However, certain paths may be blocked by EXITT due to smoke, which is evaluated at certain time intervals of the fire simulation. A different and perhaps better way to evaluate the optimal route out of

Key Words: HAZARD I, egress path, multiple objectives, dynamic programming.

the building uses multicriteria dynamic programming, which finds all Pareto optimal (nondominated) paths out of the building.

At Clemson University, researchers have been working recently on developing the theory and algorithms of fire egress using multicriteria dynamic programming.[3] As part of this work, a computer program was written to find all nondominated paths from a single node to all other nodes, considering multiple objectives that were either constants or simple, time-dependent step functions.[4] Then Kostreva and Wiecek[5] developed Algorithm Two, which allowed the use of monotone nondecreasing cost functions in the multiple objectives and found all nondominated paths from each node to every other node. The new ideas in Algorithm Two were then integrated into the framework of HAZARD I.[6] This paper compares EXITT's fire egress algorithm and Kostreva and Wiecek's Algorithm Two.

HAZARD I

HAZARD I contains three main programs of interest here : FAST, FASTplot, and EXITT. FAST, an acronym that stands for fire growth and smoke transport, is the fire simulation model. FASTplot can be used to analyze any output from FAST's fire simulation, and EXITT models fire egress based on FAST's data.

FAST is a flexible fire simulation model that requires information about the building and the source of the fire. The user must choose a building, define in detail the building's structure, the types of materials contained in the building, the ambient conditions, and the ventilation. The user then chooses where in the building and how the fire begins. Finally, the user inputs the fire simulation's duration, in seconds, to be simulated in FAST. Once all of these factors are defined by the user, FAST accesses its database (called FIREDATA) and retrieves the necessary data corresponding to the user's input. Since inputting all of this information is time consuming and tedious, the creators of HAZARD I included several predefined and illustrative scenarios that supply all the necessary information to run FAST on the defined building.

During fire simulations, FAST keeps track of the species production of carbon dioxide, carbon monoxide, concentration-time product (a mixture of toxins caused by the fire), hydrogen cyanide, hydrogen chloride, nitrogen, oxygen, soot, total unburned hydrocarbons, and water. It also keeps track of temperature, heat release, smoke levels, and many other features of fire behavior. Each room of the building, except the room of fire origin, is modeled with two distinct layers, upper and a lower, and species are measured separately on each. Within these layers, the species are considered to be uniformly distributed. The room of fire origin encompasses the upper and lower layer plus a flume layer from the initial fire. As the fire progresses, the upper layer increases in volume and the lower level decreases. FAST solves a system of differential equations by numerical integration, calculates the amounts of these species, and records them into an output file at prespecified time intervals, until the time horizon given by the user is completed. This file can then be read into FASTplot, which allows the user to make graphs and charts of the different species' productions.

EXITT,[7] the fire egress model of HAZARD I, simulates occupants' escape from the burning building. The user must create a network of nodes that mark all of the occupants' locations, possible exits, and exit routes. First, the exits must be distinguished from the other nodes and defined as doors or windows. Preference is given to doors as exits from the building, with windows chosen only if conditions are too severe to reach a door. Next, the user must define the edges between nodes where an occupant can travel, calculate the distances of the paths represented by these edges, and enter the results.

After defining all of these building network characteristics, the user places occupants at certain nodes. EXITT allows the user to give each occupant several characteristics that can affect the way he or she behaves in response to the fire. The characteristics are: age, sex, travel speed, whether or not the occupant is awake and how difficult it is to awaken them, and whether or not the occupant needs help getting out of the building. EXITT also allows the user to create fire alarms that go off at certain levels of smoke and at varying decibel levels. Once the occupants are alerted to the fire and decide to escape, EXITT uses some decision rules, including Dijkstra's shortest path algorithm,[8] to simulate a path out of the building for each occupant, given that the path is safe.

Of course, the lengths of the edges determine the total path length. If the preferred path becomes unsafe or less desirable than another path, then EXITT will choose another path. Accordingly, EXITT's algorithm first finds the shortest path from the occupant's node to the exit node, with preference given to door exits, and simulates the occupant traveling along this path. By monitoring smoke conditions and toxins and evaluating them at regular time intervals, the program can find that a certain room is suddenly an undesirable place to travel, and the occupant has to choose a new route out of the building.

EXITT evaluates smoke levels and gives rooms demerits for high levels of smoke. These demerits add lengths to any edges in, or connected to, the undesirable room. Every time demerits are added to edges, Dijkstra's algorithm reevaluates the shortest path from the occupant's node to the exit node. Thus, the path to the exit may change, or the algorithm may choose a new exit. If smoke conditions become too severe in a certain room, the algorithm can completely block off the edges by setting their lengths equal to infinity. It is possible for a person to become blocked in the building and have no means of escape.

Algorithm Two

Kostreva and Wiecek developed Algorithm Two[5] to find all nondominated paths from every node in a network to every other node in the network using multiple cost functions (time dependent and monotone nondecreasing) as the objectives.

Thus, running Algorithm Two requires a network with nodes representing locations in a building and edges representing paths from one node location to another, just as in EXITT. It also requires each edge's associated cost functions.

Users must assign each edge at least one cost function—that of travel time—and input it directly into the FORTRAN code of Algorithm Two. They may also assign additional monotone, nondecreasing cost functions to each edge. All edges, however, must have the same number and type of cost functions, as these functions accumulate along an egress path. Once all the functions are entered, Algorithm Two simultaneously minimizes the objectives to find all nondominated paths from every node to every other node.

What is a nondominated path? First, we consider a general network containing a set of nodes N = $\{1, 2, \ldots, n\}$ and a set of arcs A = $\{(i_0, i_1), (i_2, i_3), (i_4, i_5), \ldots\}$ that indicate connections between nodes. Each arc (i, j) has an associated cost vector, $C_{ij}(t)=(C_{ij1}(t), C_{ij2}(t),\ldots C_{ijm}(t))$. A path from node i_0 to i_p is the sequence of arcs P = $\{(i_0, i_1), (i_1, i_2), (i_2, i_3), \ldots, (i_{p-1}, i_p)\}$, where the first node of each arc is the same as the terminal node of the preceding arc, and each node in the path is unique. Let $C[P]_r$ represent the cost of traversing the path P, where

$$C[P]_r = \sum_{(i,j)\in P}\left[C_{ij}(t)\right]_r \tag{1}$$

for some objective r, for any of the m objective cost functions. Then path P is called a nondominated path if there exists no other path P' such that $C[P']_r <$ ($C[P]_r$ for any r, $r = 1,\ldots,m$ and $C[P'] \leq C[P]$. Thus, there may be more than one such path from one node to another node. For example, suppose the we want to find the nondominated paths from one node i to another node j in a network minimizing two objective cost functions and suppose that the following costs correspond to the three paths possible.

Let

$$w_k = \begin{bmatrix} C[P_k]_1 \\ C[P_k]_2 \end{bmatrix} \tag{2}$$

represent the cost vector for the kth path with two cost functions.

$$w_1 = \begin{bmatrix} 10 \\ 20 \end{bmatrix}, w_2 = \begin{bmatrix} 15 \\ 8 \end{bmatrix}, w_3 = \begin{bmatrix} 15 \\ 20 \end{bmatrix} \tag{3}$$

Then path one is a nondominated path since $C[P_1]_1 < C[P_2]_1$ (10<15) and

$C[P_1]_1 < C[P_3]_1$ (10<15). The first cost in path one beats the first cost in paths two and three. And, path two is nondominated since $C[P_2]_2 < C[P_1]_2$ (8<20) and $C[P_2]_2 < C[P_3]_2$ (8<20). The second cost in path two beats the second cost in paths one and three. Path three is dominated by both path one and path two since it is clear that it would be better to choose either path one or path two.

In order to compare the results of EXITT with the results of Algorithm Two, all of the data entered into EXITT from FAST and additional user input must be given to Algorithm Two. The following sections describe how this was done and what results were obtained.

Preliminaries

First we decided to specify some cost functions, assuming time travel as a given. The density of smoke visible to the human eye, which is referred to as optical density, was the first cost function we chose, based on findings that optical density is a very important factor in occupants' choices during escape.[9] We chose temperature as the next cost because occupants would easily sense it during a fire and would, of course, want to avoid areas of increased heat. We chose the level of toxins, called concentration-time product (or CT) in HAZARD I, because occupants would naturally want to avoid areas with high concentrations of toxins. However, people don't sense toxins as easily as we do optical density and temperature. No other chemical species whose production is calculated by FAST is easily sensed by humans, but we chose carbon monoxide as another objective, since it is lethal to humans in high concentration and we wanted one additional cost function.

The next specification we had to select was a suitable fire scenario. This includes the building set up, the source of the fire, the number of occupants, and occupants' locations and characteristics. Since the authors of HAZARD I provide several predefined scenarios, with all of this information encoded into HAZARD I software input files, we decided to work with one of these. However, we had to be careful in choosing a scenario, since it is important for Algorithm Two that the cost functions all be monotone nondecreasing. Therefore, we looked at the FASTplot output for the four objectives, optical density, temperature, CT, and carbon monoxide, in order to make sure that their levels during the fire were monotone nondecreasing. We chose HAZARD I's Scenario 3 because, when we looked at its output, we saw that each of these functions was monotone nondecreasing or very close to monotone nondecreasing for all rooms, for at least 200 seconds, which was long enough for the building occupants to leave the building. Figure 1 shows a floor plan layout of the ranch house used as the Scenario 3 building, which is given in the Example Cases for HAZARD I handbook. The nodes of the model are the circular nodes.

There are exits at the windows, nodes 13 and 14, and at the door, node 9. In this example, the fire starts in Room 2 and is caused by an electric heater too close to combustible bed linens. The fuel for the fire is the double bed, the bedding, and a night table whose properties can all be found in the HAZARD I fire database. The ceilings and walls are made of gypsum board, and the floors are concrete slabs. The properties of these materials are also in the HAZARD I fire database, and the Example Cases for HAZARD I Handbook[2] give more information.

After we chose the cost functions and the scenario, we ran Scenario 3 on FAST for 200 seconds to obtain all of the fire information we needed to run EXITT and the functions to run Algorithm Two. The species information was recorded every 20 seconds into an output file to be used in FASTplot. Using the species information in EXITT was easy, since HAZARD I creates a special file for this purpose. However, Algorithm Two requires time-dependent objective cost functions, and HAZARD I only produces tabular function data at certain prespecified time intervals—in this case, every 20 seconds. Thus, the data in FASTplot had to be approximated by functions. To do this, the data for the species levels in FASTplot was read into a file on a diskette and transferred onto a Macintosh computer to

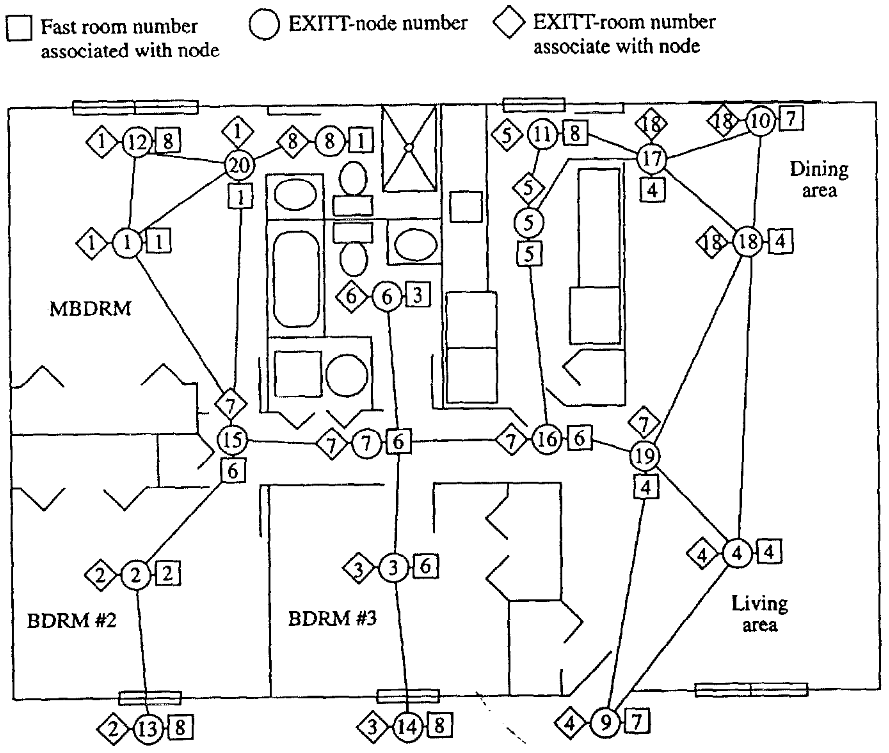

Fast room number associated with node
EXITT-node number
EXITT-room number associate with node
Dining area
MBDRM
BDRM #2
BDRM #3
Living area

Figure 1. Floor plan of house from Scenario 3.

apply the software Cricket Graph,[10] which is used to fit curves to data. Now, data for the cost objectives had to be collected for each of the house's six rooms, and separate data had to be collected for each of the four cost objectives.

Once the data was converted into a usable form, we needed to create a different function for each room of the house and for each objective using Cricket Graph. With Cricket Graph, we plotted the data on a 200-second interval time line and chose the type of curve to fit the data. In some of the data, there was a slight jump at the very beginning, due to the initial activity of the fire. For example, carbon monoxide levels increase rapidly and then level off at the start of the fire (Figure 2). After this initial rapid increase, the carbon monoxide levels grow at a steady rate for the rest of the 200 remaining seconds. Thus, when fitting curves to the data, we decided to ignore the initial burst that occurs in some of the data. In general, the curves that gave the best fit were the polynomial curves of degree three. In certain cases, however, those curves were not monotone nondecreasing on the 0 to 200 second time interval. Thus, the next best fitting type of monotone nondecreasing curve was the exponential curve. It was determined that these curves were satisfactory for our purposes. Figures 2 and 3 show some examples of curve fits from Cricket Graph.

The carbon monoxide levels in Room 6 were well-fitted with a polynomial of degree 3.

The fit of this model has an R^2 of 1.00. (Note: R^2 measures the total variation

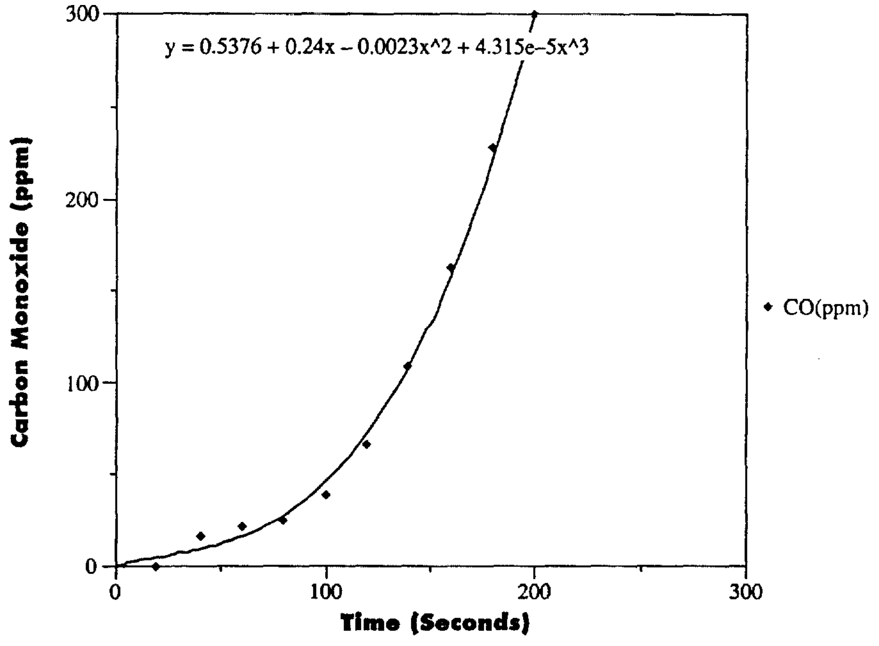


Figure 2. Carbon monoxide in Room 6.

of the y variable, which is accounted for by the relationship with x.)

The concentration time product levels in Room 4 were fitted by an exponential curve. Here the fit is not quite as good, but it is satisfactory. R^2 values are listed in the appendix for all of the objective functions. The fit of this model has an R^2 of 0.88. The R^2 values ranged from 0.66 to 1.00.

We obtained the time-dependent cost functions for each room by assigning the fitted cost functions to each arc in that room, and if an arc happened to go through two rooms, we chose one. The travel-time cost function was the only cost that differed for every arc in the network, thus, it was the last objective we addressed. Working on the assumption that every occupant would travel at a constant speed of 1.3 meters per second, which is the default in EXITT, we determined the distances of the arcs in the EXITT input file. The travel time was then computed using the simple formula, time = distance/velocity. Once all of the objectives cost functions were found, they were encoded in Algorithm Two.

Fire Egress Analysis

The next step was to run Algorithm Two, given the Scenario 3 network and its resulting objective cost functions. The network that was entered into Algorithm Two was exactly like the network in Scenario 3, except that the labels were changed and only the door was considered an exit. In a preliminary study, it became apparent that some of the nodes in Scenario 3 were unnecessary. Since

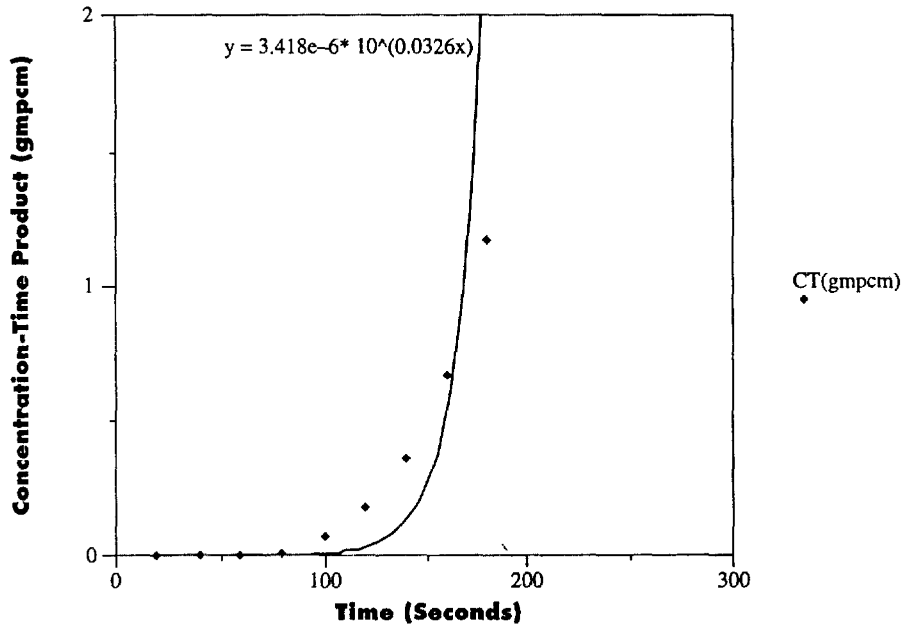


Figure 3. Concentration-time product in Room 4.

we no longer allowed the windows to be exits, the nodes and edges going to the windows were deleted. We also deleted the nodes and edges in the bathroom, since they were dead ends and no occupant would traverse those edges, unless they began in the bathroom. If an occupant did begin in a bathroom or in one of the rooms with the windows, he or she would only have to travel a very short distance (one arc) to get to the main part of the network, which the occupant would necessarily use in order to get out of the house. These nodes and edges were deleted with the understanding that they would be present in the final solution on the appropriate paths. Figure 4 is a picture of the original network setup, and Figure 5 is a picture of the modified network set p that was entered into Algorithm Two (Note: All later references to node numbers throughout the paper will be given as the Algorithm Two node numbers.)

For our comparison, we applied EXITT to Scenario 3 in HAZARD I.

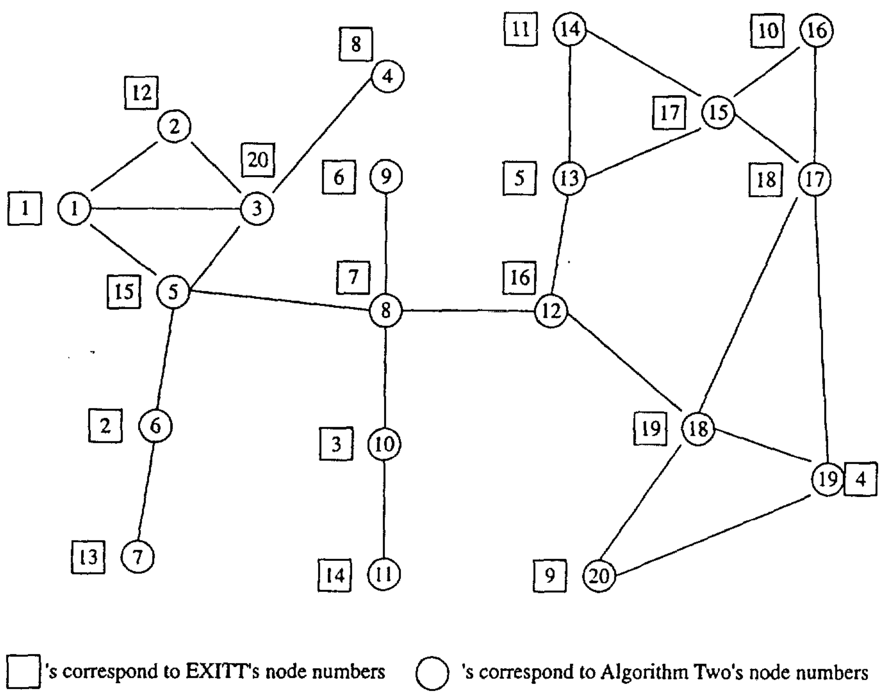
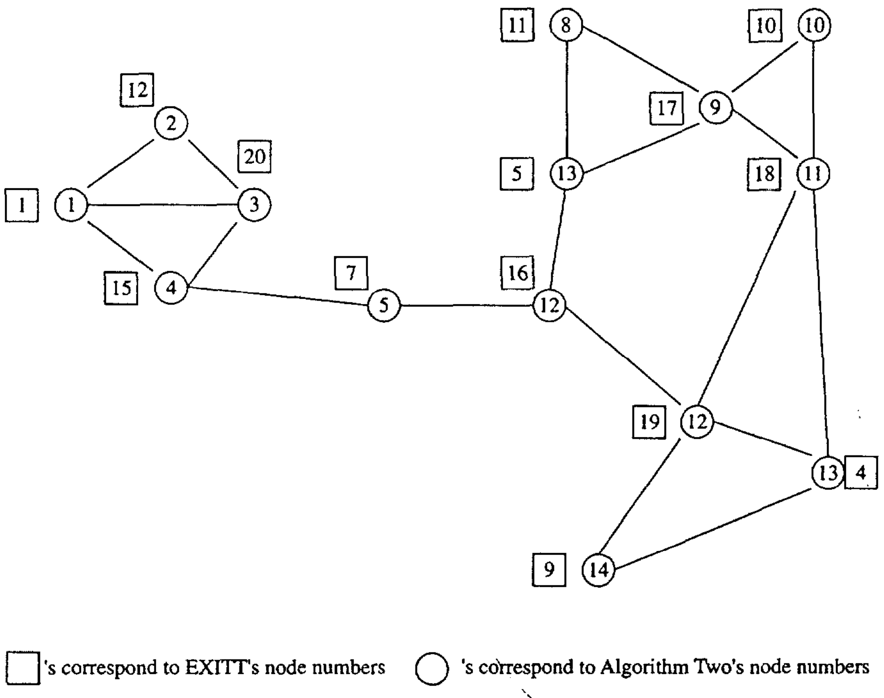

's correspond to EXITT's node numbers
's correspond to Algorithm Two's node numbers

Figure 4. The original network. Algorithm Two arcs: Room 1: (1,2), (1,3) (1,5) (2,3) (3,4), (3,5); Room 2: (5,6) (6,7); Room 3: (8,10) (10,11); Room 4: (14,15), (15,16), (15,17), (16,17), (17,18), (17,19), (18,19), (18,20), (19,20); Room 5: (13,14), (12,13), (13,15); Room 6: (8,12), (5,8), (12,18), (9,8).

Although the fire input file was ready to go, the building network in Scenario 3 had to be altered to fit the modified network we developed. This was done through the EXITT input file. However, EXITT takes into consideration many more factors about human behavior than does Algorithm Two. For example, EXITT requires detailed information about each occupant, and that information determines how the occupant will behave. For example, if the occupant is a baby, he or she will need assistance to escape. If an occupant is asleep and is a heavy sleeper, he or she will take longer to be alerted to the fire or may need to be woken up by another occupant. Also, EXITT assumes that any adults in the building will begin to investigate the source of the fire until they think the conditions demand they escape or they rescue someone. These are examples of the many choices occupants can make before they begin their escape. These choices can be found in the Technical Reference Guide for HAZARD I.[1]

Since Algorithm Two does not consider any of these characteristics, a valid comparison of the occupant characteristics to be input into Scenario 3 in EXITT had to be carefully specified. We gave each occupant characteristics to ensure



□'s correspond to EXITT's node numbers
○'s correspond to Algorithm Two's node numbers

Figure 5. Modified network. Algorithm 2 arcs: Room 1: (1,2), (1,3), (1,4), (2,3), (3,4); Room 2: none; Room 3: none; Room 4: (12,13), (11,13), (13,14), (12,14), (11,12), (10,11), (9,10), (9,11), (8,9); Room 5: (7,8) (6,7), (7,9); Room 6: (5,6), (6,12), (4,5)

TABLE 1
Results from EXITT with Node 14 as the Exit

Occupant (node)	Path	Time Began Escape (in seconds)
1	1-4-5-6-12-14	60
2	2-1-4-5-6-12-14	60
3	3-4-5-6-12-14	60
4	4-5-6-12-14	57
5	5-6-12-14	57
6	6-12-14	57
7	7-6-12-14	60
8	8-7-6-12-14	60
9	9-11-12-14	60
10	10-11-12-14	60
11	11-12-14	60
12	12-14	60
13	13-14	60

that egress would begin immediately upon being alerted to the fire. First of all, the age of each occupant was taken to be seven years old, because such occupants do not need assistance, and they are too young to investigate the source of the fire. Each occupant was also assumed to be fully awake, was given the travel speed of 1.3 meters per second, and had no disabilities that required assistance. An occupant with these characteristics was placed at each node. And for consistency, each occupant was chosen to be male.

Running EXITT produced the results in Table 1. Note that the occupant numbers corresponds with their starting nodes, and the paths are given as node-to-node paths.

Table 1 shows the results of the EXITT run, with which we compared all of our Algorithm Two runs. Most occupants began their escape at around 60 seconds, since this is when they are alerted to the fire by fire stimuli in EXITT. However, in Algorithm Two, each occupant begins escape immediately. Thus, we added 60 seconds to the travel time throughout Algorithm Two to make it comparable to the EXITT run.

Algorithm Two was tested with as many combinations of the objective cost functions as possible, with travel time always being an objective. We began by running Algorithm Two with two objectives: namely, each of the four time-dependent cost function objectives and travel time. Table 2 gives the results of the run using travel time and optical density (OD).

TABLE 2
Results from Algorithm Two with Node 14 as the Exit

Occupant (node)	Path	Time Began Escape (in seconds)
1	1-4-5-6-12-14	60
2	2-1-4-5-6-12-14	60
3	3-4-5-6-12-14	60
4	4-5-6-12-14	57
5	5-6-12-14	57
6	6-12-14	57
7	7-6-12-14	60
8	8-7-6-12-14	60
	8-9-11-12-14	
9	9-11-12-14	60
10	10-11-12-14	60
11	11-12-14	60
12	12-14	60
13	13-14	60

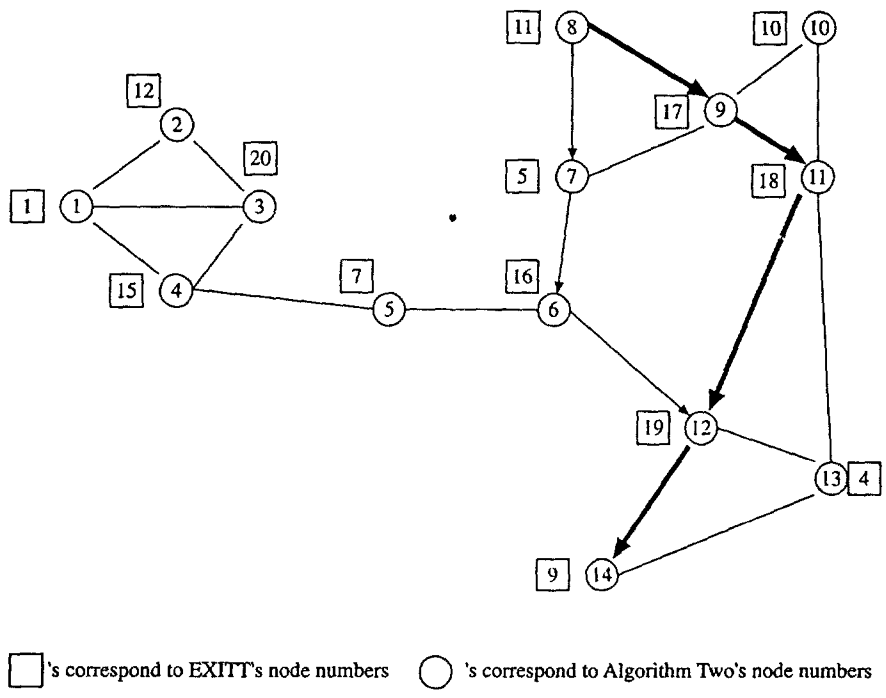
The results show all of the same paths that we got running EXITT, with an additional path from node 8 to the exit node, node 14, so that we get the paths 8 - 7 - 6 - 12 - 14 and also 8 - 9 - 11 - 12 - 14. The data from the Algorithm Two run shows that both paths are nondominated. On the first path, the total travel time is 7.6419 seconds, and the total optical density is 2.0077 1/m. On the second path, the total travel time is 8.4316 seconds, which is longer than the first path, but the total optical density is only 2.0012 1/m, which is shorter. Thus, it would be up to the occupant to decide to take the shortest route or the less smoky route. Later, we will discuss how these decisions might be made. Now, let's look at the network to see what happened.

The thin lines with arrows represent the path given in EXITT and in Algorithm Two. The thick lines with arrows represent the path given only in Algorithm Two. Since the fire begins in Room 2, all of the smoke, toxins, heat, and other products of fire end up traveling down the hallway in arcs (4,5) and (5,6) to node 6. From node 6, the species spread to the right side of the network. Thus, a higher concentration of smoke and toxins, and higher temperature outputs would occur along arcs (7,6) and (6,12). Algorithm Two finds a path that avoids these higher levels, even though the travel time is longer.

After the first Algorithm Two computation, we figured the rest with two cost functions. It turned out that, when paired with travel time, the other cost functions—that is, temperature, carbon monoxide, and CT— gave the exact same results as with travel time and optical density when run on Algorithm Two. These results can most likely be attributed to the spread of the temperature, carbon monoxide, and toxins from node 6.

Next, all possible combinations of three objectives and four objectives were run on Algorithm Two. Travel time was, of course, always one of these three objectives. It turned out that, no matter what combination we used, we always got the same paths as we did for the travel time and optimal density run. The same explanation is the bottleneck at node 6 and the location of our exit node, node 14.

Recall that Algorithm Two finds the nondominated paths from all nodes to all other nodes. Thus, the nondominated paths from all nodes to our exit node, node 14, were just a small portion of the output. Thus, there were multiple nondominated paths from certain nodes to other nodes beside what we defined as the exit node. An interesting node to examine is node 10. If we change our network and



11
8
10
10
12
2
17
9
20
5
7
18
11
1
1
3
7
16
15
4
5
6
19
12
13
4
9
14
's correspond to EXITT's node numbers
's correspond to Algorithm Two's node numbers

Figure 6: Additional path found in Algorithm 2 and not in EXITT.

TABLE 3
Results from EXITT with Node 10 as the Exit

Occupant (node)	Path	Time Began Escape (in seconds)
1	1-4-5-6-12-11-10	60
2	2-1-4-5-6-12-11-10	60
3	3-4-5-6-12-11-10	60
4	4-5-6-12-11-10	57
5	5-6-12-11-10	57
6	6-12-11-10	57
7	7-9-10	57
8	8-9-10	57
9	9-10	57
10	11-10	57
11	12-11-10	57
12	13-11-10	57
13	14-12-11-10	57

make node 10 the exit and node 14 simply a location node in the building, then we get many multiple nondominated paths to the exit. Table 3 gives the results from EXITT when node 10 is the exit, and Table 4 shows the results from Algorithm Two, which were the same for every combination of objective functions.

It can be clearly observed that two nondominated paths occur any time an occupant has to travel to node 6 on the way to the exit, node 10. The shortest path would be from 6 - 12 - 11 - 10, which takes 5.2324 seconds to traverse. However, a nondominated path was always chosen from 6 - 7 - 9 - 10 ,which takes 5.8439 seconds to traverse. This is because the path from 6 - 12 - 11 - 10 has higher levels of optical density, carbon monoxide, temperature, and CT than does the path from 6 - 7 - 9 - 10.

From this second set of calculations, we see that the consideration of nondominated paths can become very important, even for a large number of occupants in a fire. In this case, six occupants, not just one, had a pair of nondominated paths for their egress. It is possible that nearly all occupants of a building would have a set of nondominated paths, rather than the single path furnished by EXITT. Such considerations, and the differences between the nondominated paths, would form an interesting study for the building designer, fire safety engineers, and architects.

TABLE 4
Results of Algorithm Two*with Node 10 as the Exit

Occupant (node)	Path	Time Began Escape (in seconds)
1	1-4-5-6-12-11-10 1-4-5-6-7-9-10	60
2	2-1-4-5-6-12-11-10 2-1-4-5-6-7-9-10	60
3	3-4-5-6-12-11-10 3-4-5-6-7-9-10	60
4	4-5-6-12-1-10 4-5-6-7-9-10	60
5	5-6-12-11-10 5-6-7-9-10	60
6	6-12-11-10 6-7-9-10	60
7	7-9-10	60
8	8-9-10	60
9	9-10	60
10	11-10	60
11	12-11-10	60
12	13-11-10	60
13	14-12-1-10	60

*The results were the same for every combination of objective functions.

Discussion

In order to make the two methodologies operate under similar conditions, it was necessary to make some assumptions about human behavior in fires. HAZARD I contains many other parameter settings which would render it incomparable to multiple objective dynamic programming. Since models cannot handle every possible human response to fires, we work with what HAZARD I and MODP can handle simultaneously.

The travel time data in the appendix is derived from average egress speeds for the postulated occupants, given the distances in the diagram. We did not simpli-fy the travel times to one digit because that could change the results. In the inter-est of having results that can be replicated, we left the calculations with four places after the decimal point. This is important because travel times are used in all other functions.

One may question the purpose of knowing the nondominated paths rather than seeking to know the actual paths. We hold that these nondominated paths are useful for benchmark information, and as a valuable way to judge the building and how people might evacuate from a building fire. In the next section, the relationship to egress decisions is discussed.

Conclusions

Algorithm Two has an advantage over EXITT in that it can give multiple nondominated paths based on several objectives, while EXITT only finds one path based on distance (with some distance penalties based on smoke levels). Thus, Algorithm Two computations potentially allow the occupants to "choose" their objectives. For example, an occupant might want to take a longer path in order to avoid high heat, thus choosing the temperature objective as the most important objective to minimize among the nondominated solutions. Or, the occupant might want to go through a hotter room along a longer path in order to avoid extreme smoke, thus minimizing optical density among the nondominated solutions. Providing nondominated solutions reveals new decision possibilities for occupants in their egress from the building.

Some decision rules must be made for each occupant in order to ensure that one of what may be multiple nondominated paths to the exit is selected. It may be possible to decide at what levels occupants would choose one objective over another. For example, it could be determined that distance is the most important factor, but if the difference in distances is not very large, then smoke levels might be the deciding factor. However, if any objective reached a certain "high" level on a certain path, then that path would be avoided in favor of another path with a lower level of that objective. Thus, the possibilities for modeling human decision-making during a fire could be extensive and become much more realistic due to the use of multicriteria dynamic programming to find nondominated paths. Much research should be done in this area in the future.

In fact, the lack of decision-making options during fire egress in EXITT is one of EXITT's greatest weaknesses. There are many choices that the occupants can make before they begin to escape in EXITT, such as investigating the fire and assisting other occupants. However, once the occupant begins escape, her or she can make no more "decisions" based on any stimuli from the fire, except for finding the shortest path and avoiding paths with extremely high smoke visibility. EXITT considers no trade-offs, and thus, sacrifices realism.

Although it is clear that Algorithm Two is also not a realistic model of human behavior during fire egress, finding the nondominated egress paths out of a burning building from an omniscient viewpoint can be very useful for designing buildings, creating safety standards, and establishing egress paths for occupants in a building. These are some of the main objectives for creating a fire egress model, and Algorithm Two has proven to be applicable to their serious study.

References

1. Bukowski, R., Forney, C., Jones, W., Peacock, R. ,Technical Reference Guide for the HAZARD I Fire Hazard Assessment Method, NIST Handbook 146, Volume II, Center for Fire Research, National Engineering Laboratory, National Institute of Standards and Technology, US Department of Commerce, Gaithersburg, Maryland (1989).

2. Bukowski, R., Peacock, R., Example Cases for HAZARD I, NIST Handbook 146, Volume III, Center for Fire Research, National Engineering Laboratory, National Institute of Standards and Technology, U.S. Department of Commerce, Gaithersburg, Maryland (1989).

3. Kostreva, M. M., "Mathematical Modeling of Human Egress from Fires in Residential Buildings," Fire Technology, 30, (3) , (1994) pp. 338-340.

4. Wilson, M., A Time Dependent Vector Dynamic Programming Algorithm for the Path Planning Problem, Department of Mathematical Sciences, Clemson University, Clemson, South Carolina, (1992).

5. Kostreva, M. M., Wiecek, M. M., "Time Dependency in Multiple Objective Dynamic Programming," Journal of Mathematical Analysis and Applications, 173, (1), (1993) pp. 289-308.

6. Emsermann, M., Time Dependency in Multiple Objective Dynamic Programming: The General Monotone Increasing Case, Department of Mathematical Sciences, Clemson University, Clemson, South Carolina, (1993).

7. Levin, M. B., EXITT - A Simulation Model of Occupant Decisions and Actions in Residential Fires: Users Guide and Program Description, National Engineering Laboratory, Center for Fire Research, National Bureau of Standards, U. S. Department of Commerce, Gaithersburg, MD, (1987) pp.1-13, pp.25-38.

8. Dijkstra, E., ìA Note on Two Problems in Connexion with Graphs,î Numeriche Mathematics, 1, (1959) pp. 269-271.

9. Jin, T., Visibility Through Fire Smoke: Part 5, Allowable Smoke Density for Escape from Fire, Report of Fire Research Institute of Japan, No. 42. (1976) .

10. Norley, R., Rafferty, J., Cricket Graph: Presentation Graphics for Science and Business, Cricket Software, Malvern , Pensylvannia (1986).

APPENDIX, TABLE 1
Travel Times

	Travel Times		
Arc	**Travel Time (seconds)**	**Arc**	**Travel Time (seconds)**
(1,2)	1.4070	(7,9)	2.5791
(1,3)	1.4560	(8,9)	1.4829
(1,4)	2.6218	(9,10)	1.1546
(2,3)	1.2084	(9,11)	1.5772
(3,4)	3.2831	(10,11)	1.6412
(4,5)	1.6412	(11,12)	2.5361
(5,6)	1.8757	(11,13)	3.2825
(6,7)	2.1102	(12,13)	2.1582
(6,12)	1.0551	(12,14)	2.8354
(7,8)	1.6412	(13,14)	2.4054

APPENDIX, TABLE 2
Temperature Functions

Room 1:	$y=19.5847 \times 10^{0.0002371t}$	$R^2=0.83$
Room 4:	$y=18.7025 \times 10^{0.0006859t}$	$R^2=0.83$
Room 5:	$y=19.8772 \times 10^{0.00005232t}$	$R^2=0.66$
Room 6:	$y=17.8519 \times 10^{0.0024t}$	$R^2=0.98$

y=°C
t=time in seconds		

APPENDIX, TABLE 3
Optical Density Functions

Room 1:	y=0.000003811 × 10 0.0275t	R^2=0.85
Room 4:	y=0.000004985 × 10 0.0275t	R^2=0.85
Room 5:	y=0.000004985 × 10 0.023t	R^2=0.77
Room 6:	y=0.0002495 × 10 0.023t	R^2=0.66

y=1/m
t=seconds		

APPENDIX, TABLE 4
Concentration-Time (CT) Product Functions

Room 1:	$y=0.000003452 \times 10^{0.0299t}$	R^2=0.88
Room 4:	$y=0.000003418 \times 10^{0.0326t}$	R^2=0.88
Room 5:	$y=0.0000009494 \times 10^{0.0258t}$	R^2=0.83
Room 6:	$y=0.0002131 \times 10^{0.0282t}$	R^2=0.74
y= gram-minutes per cubic foot t=seconds		

APPENDIX, TABLE 5
Carbon Monoxide Functions

Room 1:	$y=0.000004345 \times 10^{0.0402t}$	$R^2=0.72$
Room 4:	$y=0.000005814 \times 10^{0.042t}$	$R^2=0.77$
Room 5:	$y=0.000000423 \times 10^{0.0387t}$	$R^2=0.79$
Room 6:	$y=05376 + 0.24t - 0.0023t^2 + 0.00004315t^3$	$R^2=1.00$

y=parts per million
t=seconds		